Chapter 10: Miscellaneous Topics for the User

In this chapter we document a variety of common operations that work differently in the Fermilab Kerberized environment.

10.1 Running Xwindows

10.1.1 UNIX

Typically, a process on a remote kerberized host isn't automatically given access to your local X display (as it is when you use **ssh**). There are a few solutions to this. One is to use the kerberized **openssh** which is now available from the KITS repository. Another is to use **kerberos** and give access with **`xauth`**, e.g.,:

```
% rsh -n -f -x <remote_host>.fnal.gov -l <username> \
  xauth add `xauth list $HOSTNAME:0`
```

(Those are backquotes around **`xauth list $DISPLAY`**.) Executing a command like this can be made more convenient. You can create an alias or shell script that sends over your **`xauth`** magic cookie (or performs an **`xhost +<remote_node>`** locally, if you use **`xhost`**, but it's considerably less safe -- someone on that host could get access to your screen and keyboard). Run it before starting the connection program. Change the script to mode 755. Here is some sample content for such a script, which we call `kxtelnet` (it's been tested between Linux, Solaris and IRIX machines):

```
#!/bin/sh

if [ $# -gt 2 -o $# -lt 1 ]; then
  echo "usage: kxtelnet RemoteHostName [RemoteUserName]" 1>&2
  exit 1
fi

host=$1
user=${2:-$USER}
```

```
case "$DISPLAY" in
  :*) disp=`hostname`$DISPLAY;;
  *)  disp=$DISPLAY;;
esac

/usr/krb5/bin/rsh -n -x -l $user $host \
    /bin/sh -c \
      "'PATH=/usr/X11R6/bin:/usr/openwin/bin:/usr/bin/X11:\$PATH; \
        export PATH; \
        xauth add `xauth list $disp`; \
        xauth list $disp'"

exec /usr/krb5/bin/telnet -x -l $user $host
```

(In the 9th line, `:*) disp=`hostname`$DISPLAY;;`, those are single backquotes around `hostname`. Same for `xauth list $disp` in 3rd to last line.) Instead of a script, you can set up an alias for a command like the following, and run it each time you restart Xwindows, before connecting to the remote host:

```
% xauth nlist <localnode>:0 | ssh <remotehost> xauth nmerge -
```

works on some machines and not others, while

```
% xauth nlist <localnode>:0 | rsh -f -x <remotehost> \
  xauth nmerge -
```

seems to work on those machines where ssh doesn't appear to work. (Often the failure is due to **`xauth`** not being in the default path.) Run this manually rather than with **`startx`** so that you can still get into Xwindows if for some reason this fails.

10.1.2 Windows NT4/98/95

If you plan to run any X applications, you'll need an X window manager. The **Reflection X Client Manager** (or other X manager, e.g., the **Hummingbird eXceed** window manager) must be running before any X client connections can be opened. You may want to place a shortcut to your X manager in **PROGRAMS > STARTUP** so that it starts automatically when you log into your PC. (And if so, it's a good idea to specify "Run: Minimized" in the shortcut properties.) We document only the **Reflection X Client Manager**.

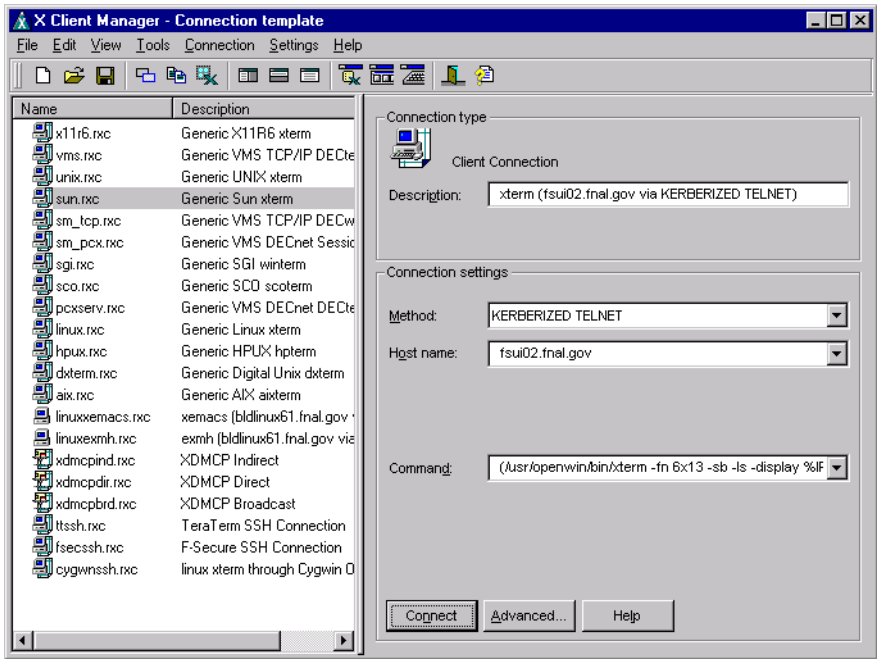
In the kerberos-users@fnal.gov mailing list archive, you can find a message containing couple of handy scripts for connecting to nodes using WRQ® Reflection X. Search for "handy scripts" and you'll find the right message![1] The first script shows how to use your own kerberos principal to log in to your

1. Message reference: Item #: 001654, Date: 01/11/03, Time: 09:14, Subject: "handy scripts for connecting to nodes using WRQ Reflection X".

own account on a remote node. The second script shows how to use the your */root* principal (see section 9.4 *Using Root Instance of your Principal*) to log in to a different account, where forwarded/renewable tickets aren't allowed.

To start **Reflection X** manually, navigate to **START > PROGRAMS > REFLECTION > REFLECTION X**. Click on it and the following screen comes up:

Run a telnet Session



The best thing to do at this point is to minimize the above window, start a **telnet** session, and run X applications from there as described in section 4.6 *Logging In Through WRQ® Reflection Software from Windows*. Once you're connected, verify that your $DISPLAY is set correctly on the remote host (at Fermilab, this should already be set for you in your UNIX login files; if it's not, check these files).

Connect Directly from X Client Manager

You can opt to connect to a host directly from the **X CLIENT MANAGER** window, *but* it does not provide encrypted connections. **Do not `kinit` from an X window!**

To connect using this window, choose a connection option from the left pane and customize it as desired, or create (and optionally save) a new connection configuration. In the right-hand pane, select `KERBERIZED TELNET` as the **METHOD** (if you leave it as just `TELNET`, the remote host will respond in portal mode). Then click **CONNECT**.

Run a telnet Session with Automatic X Application Startup

For applications that you run often, you might find it useful to configure a **telnet** connection that includes an automatic X application startup. This is described in section 19.9 *Configure X Connection to Host*. Once you have your host-specific, application-specific configurations created and saved, they should appear in the **REFLECTIONS SESSIONS** folder. To invoke, double click on the file corresponding to the host/application you want. The system will log you in and start the application in your X window manager.

If you let the **WRQ®** X client starter close the initial telnet connection after the X client starts, your remote credential cache will be destroyed. You should either copy your credential cache to another file, or check the box that keeps the initial telnet open.

Procedures for other Windows X window manager products are not documented here.

10.1.3 Macintosh

We are not recommending any particular X client for Macintosh, and the process of bringing up X windows will depend on the software used. Suggested web sites for getting information are `http://www.ncsu.edu/mac/sma/sma.html` and `http://web.mit.edu/cggriffi/www/mackerberos`/.

10.2 Usage Notes for PC's with WRQ® Reflection Installed

10.2.1 Cutting and Pasting

To cut and paste between a VT terminal window and your Windows applications using the default mouse mapping[1]:

1) Select the information in the X terminal window using the left mouse button.

2) Click the right mouse button to pop up a menu. Select **CUT** or **COPY**.

3) Click in your local application where you want to paste.

4) Click the right mouse button to pop up a menu. Select **PASTE**.

10.2.2 Using Matrix through X Windows Interface

If you use the Computing Division's **Matrix** product through the X windows interface (i.e., the software is not locally installed on your NT machine), then you must change a couple of items in the configuration. Open the **X Client Manager** (**START > PROGRAMS > REFLECTION > REFLECTION X**) and go to **SETTINGS**:

- Select **COLOR...**. In the **COLOR SETTINGS** area, change **DEFAULT VISUAL TYPE** to `PseudoColor Emulation`. Click **OK**.
- Select **FONTS...** and under **OPTIONS**, check `Allow font scaling`. Click **OK**.

10.3 Automated Processes

10.3.1 Specific-User Processes (cron Jobs)

The **kcroninit** product is provided for setting up **cron** jobs in a Kerberized environment. It gets installed automatically as part of the **kerberos** product, and as of **kerberos** v0_6, it works without **systools**. **kcroninit** creates the necessary **cron** principal and keytab file so that **cron** jobs may be authenticated

1. You can reconfigure the mouse mapping. Navigate to the **UNTITLED - REFLECTION FOR UNIX AND DIGITAL** window and find it on the **SETUP** menu.

under the user's principal. **kcroninit** can be used on each node where **cron** jobs need to be authenticated, either for AFS tokens or for remote access to other Kerberos systems.

Note that default cron ticket lifetime is picked up from the `[ftpd]` in `/etc/krb5.conf`. This is important to know if you want to increase the time limit for a cron job ticket.

For no discernible reason, many systems have been found to have permission 701 on `/var/adm`, which stops **kcroninit** from working for any user in the group to which that directory belongs. Make sure this directory is set to mode 711 or 755 before trying **kcroninit**. A later version will fix this problem automatically when encountered.

To configure a **cron** job, follow this procedure:

1) First, create the **cron** principal and keytab file. You will need to enter your Kerberos principal and password, so **you must be on a secure channel**. (The **kcroninit** program will create the new principal `<user>/cron/<host>.<domain>@FNAL.GOV` for the current user, host and domain, and will write the corresponding keytab file.) Run the commands:

```
% setup perl
% setup kcroninit
% kcroninit
```

2) Use the **`kcron`** command to initiate the **cron** jobs in an authenticated manner. Note that you will need to specify the full path to **`kcron`**, since this is not normally in your $PATH at the start of a **cron** job. In the following sample crontab entry, the command **`/home/files/myjob -(arguments as required by job)`** is authenticated as `<user>/cron/<host>.<domain>@<REALM>` (This is sufficient if authentication is needed only for access to the user's AFS files):

```
0 2 * * 0,4 /usr/krb5/bin/kcron /home/files/myjob -xyz
```

3) For access to remote systems, the `.k5login` file on the remote end must allow access to `<user>/cron/<host>.<domain>@FNAL.GOV`. If you're just creating this file, don't forget to add your `<user>@FNAL.GOV` principal, too.

To destroy the principal and keytab file (and prevent authenticated **cron** jobs from running under your account on this node), run:

```
% setup kcroninit
% kcrondestroy
```

10.3.2 Processes Running as root

If you're setting up an automated process such as a **cron** job, you have to arrange for it to get credentials when it runs. If the process is running as *root*, it is simplest, both conceptually and practically, to consider that the host on which the job runs is the party responsible for the accesses it initiates, and to have it use the `/etc/krb5.keytab` to obtain credentials as `host/<hostname>.<domain>`. To do so, first set the variable KRB5CCNAME, e.g.,:

```
% KRB5CCNAME=FILE:/tmp/krb5cc_root_$$
```

Then run **`kinit`**:

```
% /usr/krb5/bin/kinit -k host/<hostname>.<domain>
```

When you're done, get rid of the tickets:

```
% /usr/krb5/bin/kdestroy
```

10.3.3 Non-root, Non-Specific-User Processes

Here is a scheme that works for jobs that run neither as *root* nor as a specific user. This scheme provides AFS access.

First, contact the Computing Division's KDC administrator via *nightwatch@fnal.gov* and describe the job that you want to set up. Some discussion may be required to determine if this method is appropriate for your needs. If you agree to go ahead, the KDC admin will need the following information:

- a name for your job (`<jobname>`)
- the name of your division, section, or experiment (`<group>`)
- the hostnames that will need to initiate Kerberos-authenticated network connections for the job (`<hostname>.<domain>`), or ...

  the names and principals of one to three people in your group who will be the Kerberos "sub-administrators" for the job

Setting Up the Task

The KDC administrator creates a principal `<user>/<group>/<jobname>` for each Kerberos "sub-administrator", gives each one a password, and describes any extra steps that need to be taken. These principals have the authority to create, delete and change passwords for all the principals matching the pattern `<jobname>/<group>/*`.

If you're working on a farm cluster, there are certain tools available that other random systems don't have. The KDC administrator can create the special farm principal names such that when a job starts on the farm, it will have both

a Kerberos TGT as `<jobname>/<group>/farm` and an AFS token as AFS user `<jobname>`. In this case, the KDC and farm administrators take care of everything; the rest of the instructions do not apply.

The Kerberos "sub-administrators" in your group will need to:

- create a principal `<jobname>/<group>/<hostname>.<domain>` (`@REALM` is implicit) for every host which may initiate a `<jobname>` network activity.
- create a keytab file on each host containing an encryption key for the `<jobname>/<group>/<hostname>.<domain>` principal and put it in a file somewhere such that only the right UNIX id(s) on that host have access to it.

In order to create the principals and keytab files, do the following as the `<jobname>` user on each host (the **`kadmin`** commands should be issued on a single line):

```
% /usr/krb5/sbin/kadmin -p <user>/<group>/<jobname>
   Enter password:  <-- type in your password
kadmin:  addprinc -randkey
  <jobname>/<group>/<hostname>.<domain>
kadmin:  ktadd -k /path/to/keytab/file
  <jobname>/<group>/<hostname>.<domain>
kadmin:  exit
```

Then, in the home directories of the accounts which will be the targets of **`<jobname>`** activity, list all the initiator principals in a `.k5login` file as usual, e.g.,:

```
<jobname>/<group>/host1.fnal.gov@FNAL.GOV
<jobname>/<group>/host2.fnal.gov@FNAL.GOV
<jobname>/<group>/host3.fnal.gov@FNAL.GOV
```

Running the Task

In all your scripts, include a **`kinit`** command as follows:

```
% kinit -k -t /path/to/keytab/file
  <jobname>/<group>/<hostname>.<domain>
```

This must occur in the script before the script initiates a network access. (If the hostname is properly set to the full domain name, you could just use **`` `hostname` ``** in the last argument.) If you need access to AFS but your host's `/etc/krb5.conf` file does not specify `krb5_run_aklog = true` as an `[appdefault]` for **`kinit`**, then add an explicit **`-a`** flag to **`kinit`**, or run **`aklog`** as a separate step.

10.4 Sending Data from Unstrengthened to Strengthened Machines

Sending data from the strengthened realm to an unstrengthened machine is straightforward via **FTP** or an r-command. Portal mode **FTP** is available to handle sending data from an unstrengthened machine to a strengthened one.

If the strengthened target machine has a properly configured anonymous `incoming` **FTP** directory, an outside process (which can be running on an unstrengthened machine) can deposit data into it. If the target machine is *not* configured properly, the outside process can send an unauthenticated signal, e.g., an email or some other connection that signals “look for data now”, and the strengthened target machine can initiate a pull.

10.5 CVS

Different groups may implement CVS differently under Kerberos at Fermilab. Here we discuss the Computing Division’s recommended configuration which is used for its repository CDCVS. This configuration is also used by SDSS and the CD/D0/CDF Run II Code Management Working Group.

cvsh v1_4 supports Kerberized access to **CVS** repositories. **CVS** uses the *cvsuser* account. On the server side, **cvsh** must be made the default shell for the *cvsuser* account. Users must be added to that account’s `.k5login` or `.k5users` file. On the client side, users can access the CVS repository via ssh (authorized key access allowed), Kerberized rsh[1], or pserver. So if you have been accessing a repository via (nonKerberized) rsh, you’ll need to convert.

This configuration and converting to it is documented at `http://cdcvs.fnal.gov/connecting.html` and `http://www.fnal.gov/docs/products/cvs/cvs_ssh.html`. CDF users can reference `http://www-cdf.fnal.gov/offline/code_management/Dist/doc/cvsaccess.txt`.

To run a nonKerberized CVS client on a Kerberized machine, you can run two sshds:

1) the first runs on a separate IP address, allows RSA authentication, and allows only *cvsuser* to log in (*cvsuser* uses a restricted shell which

1. If you're using Kerberos rsh as the transport, and if your /etc/krb5.conf [appdefaults] says "forward = true" for rsh (or for all apps), then you have to have a forwardable ticket or create a wrapper script that does "rsh -N".

allows only CVS commands).

2) the second sshd runs on the usual IP address (but it is specified) and allows anyone to log in with Kerbreros authentication.

The CVSROOT is advertised using the IP address from item 1.